# Data Analytics Mini-Project (4-Day Sprint)

## 1. Project Overview

In this 4-day project, you will work **individually** with a real-world dataset from **Kaggle** to practice the full data analytics workflow:

- Finding and understanding a dataset
- Exploring, cleaning, and transforming data
- Building an analytical pipeline (descriptive analytics, simple models, or insights)
- Presenting your findings with **slides + live demo**

On **Friday**, each student will have:

- **10 minutes** for a slide presentation
- **10 minutes** for a live demo (notebooks / scripts / dashboard)

You are fully responsible for your own project, code, and presentation.

---

## 2. Learning Objectives

By the end of this project, you should be able to:

- Select a suitable dataset from Kaggle and understand its context.
- Design a **modular data pipeline** (exploration → cleaning → transformation → analytics).
- Apply good practices in **code organization**, **documentation**, and **version control** (Git + GitHub).

- Communicate technical work clearly through **visualizations, storytelling, and a demo**.
- Follow a **style guide** and write clear, informative **docstrings**.

---

# 3. Timeline (4-Day Plan)

**Day 1 – Project Setup & Dataset Selection**

- Choose your Kaggle dataset (see Section 4 for constraints).
- Define your **project question(s)** and goals.
- Create a GitHub repository and basic project structure.
- Start initial EDA (Exploratory Data Analysis) notebook.

**Day 2 – Cleaning & Transformation**

- Deepen EDA (missing values, outliers, distributions, correlations, etc.).
- Design and implement a **cleaning pipeline** (in scripts/notebooks).
- Start defining transformation steps (feature engineering, encoding, scaling, etc.).

**Day 3 – Analytics & Refinement**

- Implement your Descriptive analytics and visualizations.
- Refine your pipeline and modularize code (functions, classes, scripts).
- Ensure code quality, docstrings, and style.

**Day 4 – Final Polish & Presentation Prep**

- Clean up notebooks (clear dead cells / unused code, add explanations and headings).
- Finalize repository structure and README.
- Prepare and rehearse your slide deck and demo.

- Make sure everything runs **from a clean environment** (reproducibility).

**Friday – Presentations & Demos**

- 10-minute slide presentation.
- 10-minute live demo.

---

## 4. Dataset Selection (Kaggle)

You must choose a dataset from **Kaggle** that satisfies the following constraints:

- **Rows:**
    - At least **100,000 rows**
    - At most **1,000,000 rows**
- **Columns:**
    - At least **10 columns**
    - At most **30 columns**
- **Other requirements:**
    - Public, free-to-use Kaggle dataset.
    - Interesting to you (finance, health, sports, social media, environment, etc.).
    - Contains a mix of features (ideally some numeric and some categorical).

If the original dataset is larger than these limits, you may **download it and create a filtered/sample version** that meets the row/column constraints, but document clearly how you sampled it.

**Required in your project:**

- Include the **Kaggle link** in your README.

- Briefly explain in your README:
  - What is this dataset about?
  - Where does it come from?
  - What questions do you want to answer?

---

# 5. Project Requirements (Pipeline)

Your project must implement a clear pipeline with at least these stages:

1. **Exploration**
   - Inspect data types, shapes, basic statistics.
   - Explore missing values and distributions.
   - Create meaningful visualizations (histograms, boxplots, scatter plots, bar charts, etc.).
2. **Cleaning**
   - Handle missing values (drop, impute, domain-specific strategies).
   - Handle outliers (remove, cap, or justify why you keep them).
   - Fix data types and inconsistent values.
3. **Transformation**
   - Encode categorical variables when needed.
   - Scale or normalize features if relevant.
   - Create new features (feature engineering) if it helps your analysis.
4. **Analytics**
   - At minimum: strong descriptive analytics and clear visual storytelling.
   - Optional but encouraged: a simple model (e.g., regression, classification, clustering) with basic evaluation metrics.

5. **Conclusions**
    - Summarize key insights.
    - Discuss limitations and possible next steps.

---

## 6. Modularity & Project Structure

Your project must contain **modular functionalities**:

- Separate **notebooks** and **scripts** by purpose, for example:
    - `notebooks/01_eda.ipynb` – Initial exploration and visualizations
    - `notebooks/02_modeling.ipynb` – Modeling and evaluation
    - `src/cleaning.py` – Cleaning functions
    - `src/transformations.py` – Feature engineering & transformations
    - `src/utils.py` – Shared helper functions

**Suggested directory structure:**

```
project-root/
├── data/
│   ├── raw/          # Original data from Kaggle (do not modify)
│   └── processed/    # Cleaned / transformed datasets
├── notebooks/
│   ├── 01_eda.ipynb
│   └── 02_modeling.ipynb
├── src/
│   ├── cleaning.py
│   ├── transformations.py
│   └── utils.py
├── reports/
│   └── slides/       # Your final slide deck
├── README.md
└── requirements.txt  # Python dependencies
```

# 7. Deliverables

By Friday, you must deliver:

1. **GitHub Repository (per student)**
    - Clear, descriptive project name.
    - Instructor has access.
    - **Good README** (see Section 8).
2. **Code + Notebooks**
    - Modular Python scripts in `src/`.
    - Notebooks in `notebooks/` with:
        - Clean, readable structure (headings, explanations).
        - Minimal unused or commented-out code.
    - All code must run without errors from a fresh clone (assuming correct environment).
3. **Slides**
    - Final slide deck in `reports/slides/`.
    - Follows the template in Section 11 (you can adapt, but keep the core structure).
4. **Live Demo**
    - Show how your repository is organized.
    - Show how to run at least one key notebook or script.

## 8. README Requirements

Your `README.md` should include at least:

- **Project Title**
- **Short Description** (2–3 sentences)
- **Dataset**
  - Link to Kaggle dataset
  - Short description and motivation
- **Dataset Size**
  - Number of rows and columns used in your project
  - Mention if you sampled or filtered from a larger original dataset
- **Project Goals / Questions**
  - What are you trying to understand or predict?
- **Project Structure**
  - Brief explanation of folders and main scripts/notebooks.
- **How to Run**
  - Environment requirements (Python version, main libraries).
  - How to install dependencies (`pip install -r requirements.txt`).
  - How to run the main analysis (which notebooks/scripts to execute).
- **Results & Key Insights**
  - 3–5 bullet points summarizing your main findings.

## 9. Code Quality & Style Guide

We will review **code quality and style**. You are expected to:

- Use **meaningful names** for variables, functions, and classes.
- Follow a consistent style (e.g., PEP 8 for Python):
    - snake_case for functions and variables (e.g., `clean_missing_values`).
    - PascalCase for classes (e.g., `DataCleaner`).
- Avoid very long functions: break logic into smaller, reusable functions.
- Remove dead code and unnecessary print statements.
- Keep notebooks clean and narrative: mix code with explanations and headings.

## 10. Documentation & Docstrings

Every **function** and **class** in your `src/` folder must include a docstring with:

- **Short description** (one sentence)
- **Long description** (optional, 2–4 sentences if useful)
- **Args** (list of parameters, types, and meaning)
- **Returns** (type and meaning of the output)
- **Raises** (exceptions that the function may raise, if any)

**Example (Google-style docstring):**

```python
def clean_missing_values(df, strategy="mean"):
    """
    Clean missing values in a DataFrame.

    This function handles missing values according to the chosen
strategy.
    Currently supported strategies are "mean", "median", and "drop".

    Args:
        df (pandas.DataFrame): Input DataFrame with missing values.
        strategy (str): Strategy to handle missing values. One of
            {"mean", "median", "drop"}.

    Returns:
        pandas.DataFrame: A new DataFrame with missing values handled
        according to the specified strategy.

    Raises:
        ValueError: If an unsupported strategy is provided.
    """
    ...
```

You may choose Google-style, NumPy-style, or similar, but be **consistent** across the project.

---

# 11. Presentation Guidelines (Slides + Demo)

Each student has **20 minutes total**:

- **10 minutes – Slide presentation**
- **10 minutes – Live demo**

## Suggested Slide Template (10–12 slides)

1. **Title Slide**
    - Project title
    - Your name

2. **Motivation & Problem Statement**
    - Why did you choose this dataset?
    - What questions are you trying to answer?
3. **Dataset Overview**
    - Kaggle dataset link
    - Rows, columns (and confirm it meets the project constraints)
    - Main features and context
4. **Data Exploration Highlights**
    - Key descriptive statistics
    - Interesting distributions or correlations
    - 2–4 key plots with brief interpretation
5. **Data Cleaning**
    - Main data issues (missing values, outliers, inconsistencies)
    - How you handled them (drop, impute, transform, etc.)
6. **Transformations & Feature Engineering**
    - Encodings, scaling, or feature creation
    - Rationale for your choices
7. **Analytics / Modeling**
    - What types of analyses or models did you apply?
    - Key metrics or findings (accuracy, RMSE, clusters, patterns, etc.)
8. **Results & Insights**
    - 3–5 key takeaways
    - What did you learn from the data?
9. **Limitations**

    - What are the limitations of your data or approach?
    - What would you improve with more time?

10. **Conclusion & Next Steps**

- Summary of the project
- Ideas for future work or extensions

11. **Live Demo Intro** (optional)

- A quick slide explaining what you will show in the demo.

During the **demo**, show:

- How your repository is organized.
- How to run at least one key notebook or script.
- A highlight of one or two important visualizations or results.

---

# 12. Evaluation Criteria (High-Level)

You will be evaluated on:

1. **Technical Work (Pipeline)**
    - Quality of exploration, cleaning, transformation, and analytics.
    - Correct and meaningful use of methods.
2. **Code Quality & Structure**
    - Modularity, readability, style, and organization.
    - Proper use of functions, classes, and docstrings.
3. **Reproducibility**
    - Clear instructions in README.
    - Code runs without errors on a fresh clone (within reasonable setup).

4. **Analysis & Insight**
    - Depth of understanding of the data.
    - Clarity and relevance of insights / conclusions.
5. **Communication**
    - Slide design, structure, and clarity.
    - Quality of explanations during the presentation and demo.
    - Ability to answer questions about your pipeline and choices.

---

# 13. Collaboration & Academic Integrity

This is an **individual** project.

- You may:
    - Discuss high-level ideas or challenges with classmates.
    - Use online resources (documentation, tutorials, etc.).
- You must not:
    - Copy another student's code or notebooks.
    - Present external work as if it were entirely your own.

Always:

- Cite any tutorials, blogs, or code snippets you used or adapted (mention them in README or notebook comments).

## 14. Checklist Before Friday

- □ Kaggle dataset chosen that satisfies size constraints (100k–1M rows, 10–30 columns).
- □ GitHub repo created and shared with instructor.
- □ Project structure organized (data/, notebooks/, src/, reports/).
- □ Cleaning and transformation scripts implemented with docstrings.
- □ At least one EDA notebook and one analytics/modeling notebook.
- □ README completed with how-to-run instructions and key insights.
- □ Slides finished following the template.
- □ Demo tested: can you run key steps from scratch?